University of Westminster
Department of Computer Science

ECWM511 Coursework 2 (Semester 2)

Module leader	Dr D. Dracopoulos
Unit	Coursework 2
Weighting:	50%
Qualifying mark	30%
Description	
Learning Outcomes Covered in this Assignment:	LO1, LO2, LO3, LO4, LO5
Handed Out:	23/2/2015
Due Date	23/3/2015 10:00am
Expected deliverables	Source code/XML files
Method of Submission:	Online via Blackboard
Type of Feedback and Due Date:	Individual feedback verbally straight after the viva and written individual feedback via Blackboard within 2 weeks of submission **All marks will remain provisional until formally agreed by an Assessment Board.**

Assessment regulations

Refer to section 4 of the "How you study" guide for undergraduate students for a clarification of how you are assessed, penalties and late submissions, what constitutes plagiarism etc.

Penalty for Late Submission

If you submit your coursework late but within 24 hours or one working day of the specified deadline, 10 marks will be deducted from the final mark, as a penalty for late submission, except for work which obtains a mark in the range 40 – 49%, in which case the mark will be capped at the pass mark (40%). If you submit your coursework more than 24 hours or more than one working day after the specified deadline you will be given a mark of zero for the work in question unless a claim of Mitigating Circumstances has been submitted and accepted as valid.

It is recognised that on occasion, illness or a personal crisis can mean that you fail to submit a piece of work on time. In such cases you must inform the Campus Office in writing on a mitigating circumstances form, giving the reason for your late or non-submission. You must provide relevant documentary evidence with the form. This information will be reported to the relevant Assessment Board that will decide whether the mark of zero shall stand. For more detailed information regarding University Assessment Regulations, please refer to the following website:**http://www.westminster.ac.uk/study/current-students/resources/academic-regulations**

ECWM511 MOBILE APPLICATION DEVELOPMENT - Assignment 2
Deadline 23/3/2015, 10:00am

Dr Dimitris C. Dracopoulos
Email: d.dracopoulos@westminster.ac.uk

Description

You are required to implement an Android application described by the specifications below. The application will be a basic *appointment management* application, which the user will be using to create appointments with full details describing what is involved in each appointment.

It is important to follow exactly the specifications and your implementation must conform to these:

1. When the application starts, it presents the user with a calendar displaying the dates of the current month and year and 6 buttons labelled *Create Appointment*, *View/Edit Appointments*, *Delete Appointment*, *Move Appointment*, *Search* and *Translate Appointment*.

   The user should be able to select a day by clicking on the corresponding date of the calendar. The user can select a different month or year by using the displayed calendar.

   (5 marks)

2. After the user selects a date, clicking on the *Create Appointment* button displays the user with 3 textboxes and a *Save* button. The 3 textboxes will prompt the user to enter three separate things: The title of the appointment event (e.g. Meeting with John), the time of the event and the details of the appointment (e.g. we are going to discuss, this and that and that...). The application should allow the user to enter text of arbitrary length for the details of the appointment and scrollbars should be added to the textarea displaying them, so that the user can scroll up and down and edit all the information entered.

   Pressing the *Save* button will save the title and details of the appointment in an SQLite table. The table should contain separate columns for the date of the appointment, time, title of the appointment and details of the appointment (with any additional necessary columns left up to you to create).

   (20 marks)

3. Any number of appointments could be created for a single date in the previous subquestion. However, any appointment title should be unique for a single date (even if the details and/or time of the 2 appointments are different). I.e. if the user has selected 2nd of March 2012 and he/she attempts to enter 2 appointments with the tile "Meeting with

John", a pop up error window should be displayed "Appointment Meeting with John already exists, please choose a different event title".

Two appointments with the same title are allowed if they occur on different dates.

(5 marks)

4. After the user selects a date in the initial screen, clicking on the *Delete Appointment* button displays the user with 2 options as buttons: "Delete all appointments for that date" which will delete from the database all the corresponding appointments and "Select appointment to delete" button which will display all of the appointments titles on that date, sorted in ascending order by time and according to the following format:

```
1. 15:00 Meeting with John
2. 17:00 Team meeting
3. 20:00 Party at Helen's
```

The user will be asked to enter a number e.g. 2, and then a pop up window will ask for a confirmation *Would you like to delete event: "Team Meeting"?*, with 2 buttons labelled "Yes" and "No". After pressing the "Yes" button, the event will be deleted from the database.

(20 marks)

5. After the user selects a date in the initial screen, clicking on the *View/Edit Appointment* button displays the user with all appointments of that date (in a similar format with the previous subquestion) and allows him/her to edit all the details of a chosen appointment (time, title, details). All the updated information is persisted in the database.

(6 marks)

6. After the user selects a date in the initial screen, clicking on the *Move Appointment* button displays the user with all appointments for that date, asking him/her to select a specific appointment. Then the application should allow the user to select another date from a calendar view, in which case the appointment is transferred automatically to the newly selected date (the database is updated accordingly).

(10 marks)

7. After the user selects a date in the initial screen, clicking on the *Translate Appointment* button allows the user to select an appointment and then translate the details of the appointment (not the title, just the details) to any language of his/her choice using the Microsoft Translate API techniques introduced in the lectures and tutorials. The source language and the target language should be displayed in 2 separate drop down buttons.

Once the user does the translation, the translated contents of the details of the appointment are displayed in the target language and the user is given the choice to save the translated version of the appointment (previous contents of the appointment details will be deleted from the database).

(10 marks)

8. Add an option "Select Languages" which will allow the user to select which languages he/she like to appear in the drop down buttons of the previous subquestion. The user

should be able to add/delete a language assuming that it is in the list of languages supported by the Microsoft Translate API and thus customise the source/target language drop down buttons according to his/her preferences.

(6 marks)

9. The *Search* button of the initial screen will be used by the user in order to retrieve appointments for which he/she does not remember the date. Clicking on the button will ask the user to enter a string used for the search. The application will search all future appointments and try to match the entered string with the title and details of any appointment(s). The matched appointment(s) will be displayed to the user and he/she will be able to select one of them to see the full details of it.

   E.g. if the user enters "Wood" then an appointment with title "Meeting with Woodworth" would be matched and displayed. Also an appointment with details "I am going to see Woodworth to see what he thinks of the new programming language I came up with" would be matched and also displayed.

   Search should be case insensitive and the user is allowed to enter any string including the case which a string contains parts of 2 words separated by a space character. E.g. a search string "see Wood" would display the second appointment above. Exact search of the entered string is thus required.

   The search implementation must be done programmatically using the Java language and must NOT be done using SQL statements to extract appointment rows matching the entered search string. SQL statements for this subquestion can be used only to extract ALL appointment rows occurring in the future which are then manipulated further using Java code.

   (8 marks)

Marking Scheme: The marks achieved for each part of the program are indicated in the description of the task above. In addition to these the following will be taken into account:

- *Code readability* (structure, comments, variable naming, etc,): 5%
- *Implementation* (e.g. quality, efficiency, look and feel of the application, based on fonts, colours, etc.): 5%

The maximum for work which does not compile (or XML files with syntax errors causing the Java code not to compile) is 30%.

Based on the functionality implemented, the marks awarded will consist of 2 parts:

- 30% of the marks achieved will be awarded based on the submission.
- The remaining 70% of the marks for the implementation will be awarded after a compulsory viva, that will test the understanding of the code by the student. The student will be asked to demonstrate the application and will be asked questions about the code to demonstrate his/her understanding.

A compulsory viva for each student based on his/her submission will take place during the tutorial sessions in the week beginning the 23rd of March 2015. Students

should attend the tutorial slot which is in their FORMAL timetable and they will NOT be allowed to attend a viva in a different tutorial slot. Failure to turn up in the viva on the specified slot below will result in awarding only 30% of the marks achieved for the submission (see marking scheme above)

Students who have their formal tutorial on Monday should come to the corresponding slot (11:00-13:00) on Monday 23rd of March. Students who have their formal tutorial on Tuesday should come to the corresponding slot (16:00-18:00) on Tuesday 24th of March. Students who have their formal tutorial on Friday should come to the corresponding slot (11:00-13:00 or 16:00-18:00 depending on what is in your timetable) on Friday 27th of March.

It is the responsibility of each student to make sure that during the viva the code runs properly in the lab used during the viva, i.e. you should make sure in advance (allow enough time before the viva day) that everything is running properly in the machine you will be using. If you developed code at your home computer, it is your responsibility that you port it to the lab in advance, before the viva. Marks will be awarded based on the demo/viva and excuses of the type "it used to run - don't know what happened since last time" will not be accepted or awarded with extra marks.

You are allowed to use your own laptop during the viva if you wish to.

Submission of assignments using a different method other than Blackboard will not be accepted and zero (0) marks will be awarded in such cases.

Deadline: Monday 23rd of March 2015, 10:00am.

Submission Instructions

Files to submit: All of the files of the Eclipse project of your application in a zip file.

You should submit via BlackBoard's Assignment functionality (do NOT use email, as email submissions will be ignored.), all the files described above. A single zip file containing all the above files could be submitted alternatively.

Note that Blackboard will allow to make a submission multiple times. Make sure before submitting (i.e. before pressing the `Submit` button), that all the files you want to submit are contained there (or in the zip file you submit).

In the case of more than one submissions, only your last submission before the deadline given to you will be marked, so make sure that all the files are included in the last submission attempt and the last attempt is before the coursework deadline.

Request to mark submissions which are earlier than the last submission before the given deadline will be ignored as it is your responsibility to make sure everything is included in your last submission.

The following describes how to submit your work via BlackBoard:

1. Access `https://learning.westminster.ac.uk` and login using your username and password (if either of those is not known to you, ask the HelpDesk at the Library.).

2. Click on the module's name, `MODULE: ECWM511.2015 MOBILE APPLICATION DEVELOPMENT` found under `My Modules & Courses`.

3. Click on the `Assignments` button found on the left hand side menu.

4. Click on `View/Complete Assignment`.

5. Attach your zip file containing all your files of your Android project, by using the `Browse` button.

6. Fill in the requested information:

   - *Comments*: Type your full name and your registration number, followed by:
     "I confirm that I understand what plagiarism is and have read and understood the section on Assessment Offences in the Essential Information for Students. The work that I have submitted is entirely my own. Any work from other authors is duly referenced and acknowledged."

7. Click the `Submit` button.

If Blackboard is unavailable before the deadline you must email `FSTRegistry@westminster.ac.uk` with **cc:** to myself and your personal tutor before the deadline with a copy of the assignment, following the naming, title and comments conventions as given above and stating the time that you tried to access Blackboard. You are still expected to submit your assignment via Blackboard. Please keep checking Blackboard's availability at regular intervals up to and after the deadline for submission. You must submit your coursework through Blackboard as soon as you can after Blackboard becomes available again even if you have also emailed the coursework to the above recipients.

Coursework Marking scheme

The Coursework will be marked based on the following marking criteria:

Criteria	Mark per component	Mark provided	Comments
Implementation	**100**		
Functionality	90		For a split of the marks see the subquestions description in the main description of the coursework
Code Readability	5		structure, comments, variable naming, etc,
Software Quality	5		Quality, efficiency, etc.
Total	**100**		